

Compte- rendu PTI #05

Cette quatrième PTI couvre le domaine du déploiement d'infrastructures de gestion de clés publiques en Intranet.

Objectifs

Les certificats numériques permettent à des utilisateurs de signer numériquement des messages (e-mails, documents administratifs numériques, etc ...) et de s'authentifier sur des sites web sécurisés ou des tunnels VPN ; ils permettent également à des machines (serveurs web, passerelles VPN, etc ...) de confirmer leur identité pour éviter tout détournement de flux de données.

Une PKI a pour fonction de fournir un service de création, puis de gestion de certificats numériques au sein d'une organisation.

Compétences

Cette PTI couvre les compétences suivantes :

- **C21 - Installer et configurer un micro- ordinateur**
- **C22 - Installer et configurer un réseau**
- **C23 - Installer et configurer un dispositif de sécurité**

Outils utilisés

Au cours de la réalisation de la PTI, les outils suivants ont été utilisés :

Matériel :

- Micro- ordinateur (serveur)

Logiciels :

- Système d'exploitation GNU/Linux Gentoo 2004.3- r1 (noyau 2.6.11)
- Serveur web Apache (version 1.3.33) + mod_ssl (version 2.8.22)
- Cryptographie OpenSSL (version 0.9.7e)
- Certification PHPKI (version 0.60, modifiée pour les besoins de l'AP)
- Scripting PHP (version 5.0.3)

Couche cryptographique : OpenSSL

Note : Nous compilerons tous les logiciels à partir de leur code source de manière à activer des options particulières si besoin, comme par exemple le support de SSL par Apache.

Toutes les opérations cryptographiques (génération de clés asymétriques, chiffrement, signature numérique...) réalisées côté serveur par la PKI sont assurées par OpenSSL, une collection d'utilitaires libres et de bibliothèques de fonctions qui fournissent une couche cryptographique aux systèmes UNIX/BSD.

Voici la méthode employée pour installer OpenSSL à partir des sources du logiciel, en suivant la procédure habituelle de compilation sous les OS libres :

```
$ tar xvzf openssl-0.9.7e.tar.gz
$ cd openssl-0.9.7e/
$ ./configure
$ make
$ make test
$ make install
```

Vérifions que l'application soit bien installée et fonctionne correctement :

```
$ openssl version
```

```
OpenSSL 0.9.7e 25 Oct 2004
```

Serveur HTTP(S) : Apache + mod_ssl + PHP

Pour pouvoir accéder depuis n'importe où aux services fournis par la PKI, nous avons besoin d'une interface web qui sera garantie par le serveur HTTP libre Apache.

Mais avant de compiler Apache nous devons préparer le module de chiffrement des communications HTTP entre l'interface web de la PKI et les utilisateurs / administrateurs. En effet, le protocole HTTP n'est pas sécurisé et il est possible d'intercepter le contenu des échanges comme les mots de passe ou pire encore, les clés générées par la PKI.

mod_ssl est un module pour Apache qui ajoute une couche de chiffrement SSL (HTTPS) - et maintenant TLS⁽¹⁾ - de façon à créer un tunnel virtuel sécurisé entre les 2 entités communicantes, généralement un client et un serveur HTTPS.

```
$ tar xvzf apache_1.3.33.tar.gz
$ tar xvzf mod_ssl-2.8.22-1.3.33.tar.gz
$ cd mod_ssl-2.8.22-1.3.33/
$ ./configure --with-apache=../apache_1.3.33
```

A ce stade, les sources d'Apache ont été modifiées par la commande précédente de manière à leur ajouter le code de mod_ssl à compiler en même temps que les autres modules du serveur.

Voici la procédure suivie pour installer le serveur web et créer le certificat numérique du serveur web :

```
$ cd ../apache_1.3.33
$ SSL_BASE=../openssl-0.9.7e/ ./configure --prefix=/web/apache\
  --enable-module=most --enable-module=ssl --enable-shared=ssl
$ make
$ make certificate
$ make install
```

Il convient à présent de modifier le fichier de configuration du serveur web (httpd.conf) :

```
ServerName bouh
DocumentRoot "/web/www/"
```

Note : il existe dans /etc/hosts une entrée "bouh" associée à l'adresse IP du serveur.

On vérifie que le serveur se lance correctement :

```
$ /web/apache/bin/apachectl startssl
Apache/1.3.33 mod_ssl/2.8.22 (Pass Phrase Dialog)
Some of your private key files are encrypted for security reasons.
In order to read them you have to provide us with the pass
phrases.

Server bouh:8443 (RSA)
Enter pass phrase:

Ok: Pass Phrase Dialog successful.
/web/apache/bin/apachectl startssl: httpd started
```

Il ne reste plus que PHP à installer, qui servira à automatiser le traitement de certaines opérations de la PKI, notamment l'interfaçage avec OpenSSL. De même que pour toutes les applications précédemment installées nous allons le compiler sous forme de module pour Apache à partir de ses sources :

```
$ tar xvjf php-5.0.3.tar.bz2
$ cd php-5.0.3/
$ ./configure --prefix=/web/php --with-apxs=/web/apache/bin/apxs
--enable-ssl
$ make
$ make install
```

Nous devons encore une fois éditer le fichier de configuration d'Apache pour lui préciser de charger le module PHP au démarrage :

```
LoadModule php5_module          libexec/libphp5.so
AddModule                       mod_php5.c
AddType application/x-httpd-php .php .php4 .php3
```

On crée maintenant un petit script PHP `phpinfo.php` censé afficher les paramètres de l'interpréteur pour tester son bon fonctionnement :

```
<?php
    phpinfo();
?>
```

Pour achever la vérification, on ouvre un navigateur (au hasard : Mozilla Firefox ;) et on le pointe vers l'adresse <https://bouh:8443/phpinfo.php> : une fenêtre s'ouvre en nous demandant d'accepter le certificat numérique du serveur. Une fois vérifié et accepté, on constate que nous sommes bien dans un tunnel SSL (apparition du petit cadenas fermé dans la barre d'adresses) et que la page d'info de PHP s'affiche.

Solution de PKI : PHPKI

Maintenant que tous les outils nécessaires au bon fonctionnement de la PKI sont installés et fonctionnent correctement, il ne reste plus qu'à installer la PKI à proprement parler.

La solution de PKI retenue est **PHPKI**⁽¹³⁾, un *front-end* (une surcouche) pour OpenSSL codé en PHP. La finalité de mon projet étant de fournir un service de génération de certificats électroniques que ce soit pour des utilisateurs (signature de données/e-mails) ou bien pour des serveurs (authentification VPN/serveurs SSL), le tout gratuitement et en utilisant uniquement des logiciels libres.

ATTENTION : Lors de ma première installation de PHPKI (parce qu'il y en a eu beaucoup ;) j'ai été confronté à quelques erreurs d'écriture sur les fichiers d'installation, et par conséquent l'application, puisque mal installée, fonctionnait peu voire pas du tout. Après avoir passé quelques jours à déboguer les sources, il est apparu que PHPKI ne peut fonctionner si le "Safe Mode" de PHP est activé et si les "Register Globals" (la portée des variables globales) sont désactivées ; j'ai donc dû adapter la configuration de PHP en conséquence, sachant toutefois que la modification de ces options sont des failles potentielles et connues du public.

Après avoir récupéré les sources sur le site officiel, on procède à l'installation comme illustré par la copie d'écran suivante :

https://bouh:8443/phpki-0.60/setup.php

PHPKi: S/MIME Certificate Authority

PHPKi S/MIME Certificate Authority

[ReadMe](#) [Setup](#) [About](#)

Certificate Authority Initial Setup

Root Certificate Data	
Organization *	Alternatives 87
Department/Unit *	PKI Alternatives 87
Common Name * This is embedded in certificates, and is most often displayed in e-mail clients as the <i>Issued By</i> text. This is usually the full name of your certificate authority (i.e. ACME Certificate Authority).	Autorité de Certification Alternatives 87
Technical Contact E-mail Address * Enter an e-mail address where users should send correspondence regarding your certificate authority and the certificates you issue.	pki@alternatives87.eu.org
Locality *	Limoges
State/Province *	FR
Country *	FR
Password * This password will be used to protect your root certificate private key. Do not lose or forget this password.	***** Again *****
Certificate Life * Enter the number of years you wish your root certificate to be valid.	5 Years
Certificate Authority Base URL Enter the public Web address where your Certificate Authority will reside. The address should end with a trailing slash (/) character. This address will be embedded in all certificates issued by your CA, for informational purposes.	https://bouh:8443/pki/

Configuration Options	
Location of OpenSSL Executable * Enter the location of your OpenSSL binary. The default of /usr/bin/openssl is usually ok.	/usr/bin/openssl
File Upload Prefix This is an optional prefix which will be added to root certificate and certificate revocation list file uploads. Normally the root certificate is uploaded as caroot.crt. With a prefix like "acme_", the root certificate would be uploaded as "acme_caroot.crt".	alt87_
Page Header Title This title will be displayed superimposed over the PHPKi logo at the top of every page.	PKI Alternatives 87
Help Document Contact Info This text will be inserted into the online help document under the "Getting Additional Help" section. Include full contact info for the convenience of your users. Use HTML tags to improve presentation.	<pre> Contact:
 PKI Alternatives 87
 Alternatives 87
 87000 Limoges

 E-mail: pki@alternatives87.eu.org is preferred.</i>
 </pre>

* Required field

PHPKi v0.60 - Copyright 2003 - William E. Roadcap

Exploitation du système

Selon le principe d'une PKI, PHPKI est séparé en entités distinctes, mais moins bien délimités que le modèle théorique à savoir que l'application remplit bien les rôles d'autorité de certification (CA), d'enregistrement (RA), de stockage (Repository) de séquestre mais pas d'entité finale. De plus, la PKI est ici scindée en deux parties :

- l'une publique où les utilisateurs peuvent télécharger le certificat de la CA, la dernière CRL et tous les certificats qui ont été délivrés par la CA;
- l'autre est privée et son accès est restreint au niveau du serveur où l'administrateur de la PKI peut créer les certificats demandés par les utilisateurs, les renouveler et les révoquer.

Dans un premier temps nous allons jouer le rôle de l'administrateur en créant un nouveau certificat numérique pour notre serveur HTTPS.

Toutes les opérations de pilotage de la PKI sont à effectuer avec un navigateur web. Nous nous rendons donc dans la section prévue à cet effet et nous remplissons les champs avec les informations requises; il est ensuite demandé de confirmer les informations fournies avant de valider la création.

Une fois validé, le certificat est généré côté serveur par OpenSSL (qui reçoit ses paramètres par PHPKI) puis il est demandé d'enregistrer le certificat sur notre machine.

Maintenant que nous avons notre certificat (... et sa clé privée, le format PEM fusionnant ces deux éléments dans un seul fichier). On copie donc le fichier téléchargé avec les autres certificats du serveur web et on édite son fichier de configuration :

```
SSLCertificateFile /web/apache/conf/ssl.crt/www.alternatives87.eu.org.pem
SSLCertificateKeyFile /web/apache/conf/ssl.crt/www.alternatives87.eu.org.pem
```

On redémarre le serveur HTTPS...

```
$ /web/apache/bin/apachectl stop

/web/apache/bin/apachectl stop: httpd stopped

$ /web/apache/bin/apachectl startssl

Apache/1.3.33 mod_ssl/2.8.22 (Pass Phrase Dialog)
Some of your private key files are encrypted for security reasons.
In order to read them you have to provide us with the pass
phrases.

Server bouh:8443 (RSA)
Enter pass phrase:

Ok: Pass Phrase Dialog successful.
/web/apache/bin/apachectl startssl: httpd started
```

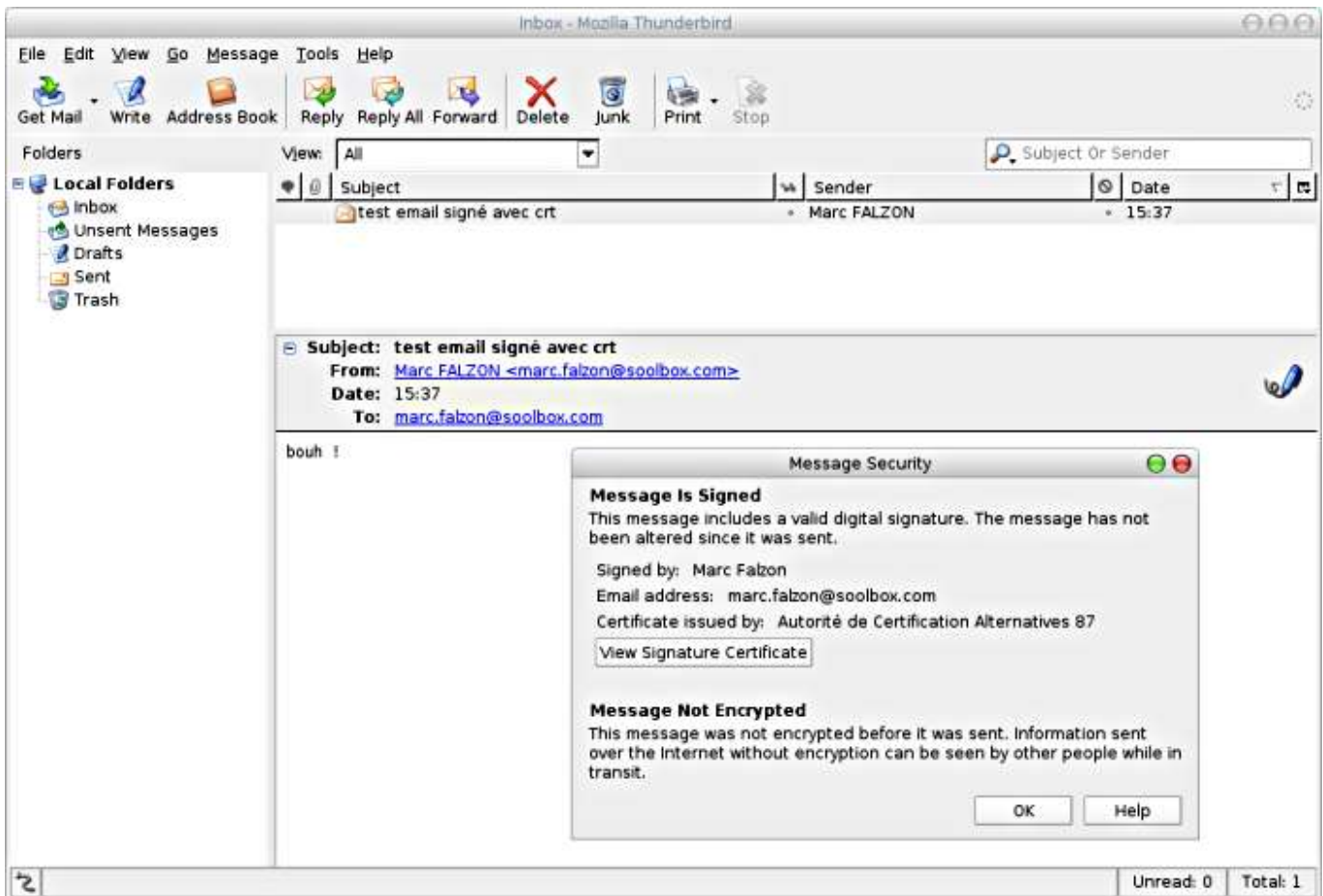
...et on retourne à notre navigateur pour vérifier si le nouveau certificat est disponible :



Maintenant, imaginons que nous recevons une demande d'un utilisateur nécessitant un certificat numérique pour signer ses e-mails : une fois dans la section adéquate nous suivons la même procédure que pour un certificat de site web, à la différence qu'on demande cette fois une adresse e-mail et non plus une adresse web.

Une fois le certificat téléchargé (format PKCS12 pour les certificats utilisateurs), on lance notre client mail (ici Mozilla Thunderbird) et on importe notre nouveau certificat; après avoir rappelé le mot de passe entré lors de la génération du certificat au niveau de la CA le logiciel nous informe que ce dernier est convenablement intégré à la base de données de certificats interne. Il faut également importer dans le client mail le certificat de la CA sinon les certificats ne seront pas valides (*not trusted*) : ceci fait on peut désormais envoyer et recevoir des e-mails dignes de confiance si on a importé le certificat du correspondant.

Ci-dessous une copie d'écran d'un e-mail signé envoyé à moi-même avec le client mail Thunderbird de Mozilla :



En cliquant sur "View Signature Certificate" on constate en effet que l'e-mail que j'ai reçu provient bien de moi-même :

Si un certificat délivré par notre CA venait à être perdu ou volé il est impératif que son propriétaire nous en informe dans les plus brefs délais afin que l'administrateur puisse le révoquer et mettre à jour la CRL.

Pour l'exemple, nous imaginons qu'un pirate se soit infiltré sur le serveur d'Alternatives 87 et ai copié son certificat et sa clé privée; l'administrateur du serveur s'en aperçoit (trop tard) et nous demande de révoquer le certificat actuel : pour ce faire l'administrateur de la PKI se rend dans la section de gestion où il contrôle la "vie" des certificats qu'il a délivrés.

Après révocation du certificat et confirmation de sa révocation, il faut à présent régénérer la CRL pour avertir les utilisateurs de ne plus faire confiance à ce certificat : nous devons nous rendre dans la section (non visible par défaut) de génération de CRL. Ceci fait, il faut importer la nouvelle CRL dans le navigateur (et il appartient AUX UTILISATEURS de mettre à jour fréquemment leur CRL).

Voici une copie d'écran de liste des certificats générés par notre CA; on le certificat que nous venons de révoquer apparaît en rouge :

PHPKi
PKI Alternatives 87

Menu Public Policy Help About

CERTIFICATE MANAGEMENT CONTROL PANEL

Search: Valid Revoked Expired

Status	Issued	Expires	User's Name	E-mail	Organization	Department	Locality	
Revoked	05-mar-31	06-mar-31	www.alternatives87.eu.org	web@alternatives87.eu.org	Alternatives 87	Site web Alternatives 87	Limoges	<input type="button" value="Display"/> <input type="button" value="Renew"/>
Valid	05-mar-31	06-mar-31	Marc Falzon	marc.falzon@soolbox.com	Soolbox	Soolbox	Paris	<input type="button" value="Display"/> <input type="button" value="Download"/> <input type="button" value="Revoke"/> <input type="button" value="Renew"/>

PHPKi v0.60 - Copyright 2003 - William E. Roadcap

Désormais toute tentative de connexion sécurisée avec le serveur web utilisant ce certificat se soldera par ce message :

